. 86 : PNW- 67

USDA Forest Service
General Technical Report PNW-67 1978

programs for road network planning

Ward W. Carson
Dennis P. Dykstra

RETURN TO GOV. DOCS. CLERK

Abstract

This paper describes four computer programs developed to assist logging engineers to plan transportation in a forest. The objective of these programs, to be used together, is to find the shortest path through a transportation network from a point of departure to a destination. Three of the programs use the digitizing and plotting capabilities of a programable desk-top calculator system to conveniently enter and store a numerical description of the network to be analyzed. The fourth program solves the problem by an efficient optimization algorithm. The BASIC language programs are described through an example problem, and complete listings are provided.

KEYWORDS: Road administration/planning (forest), transport system design (forest), computer programs/programing, computers (desk-top).

The use of trade, firm, or corporation names in this publication is for the information and convenience of the reader. Such use does not constitute an official endorsement or approval by the U.S. Department of Agriculture or by Oregon State University of any product or service to the exclusion of others which may be suitable.

AUTHORS

Ward W. Carson is Mechanical Engineer, Pacific Northwest Forest and Range Experiment Station, Seattle, Washington. Dennis P. Dykstra is Assistant Professor of Forest Engineering, Oregon State University, Corvallis, Oregon.

Contents

Introduction

The determination of the shortest route (or path) through a network of available routes is often an important step in planning transportation. Although problems in finding the shortest route frequently arise in other disciplines (Bradley 1975), they are most commonly associated with road networks, and this paper will discuss these problems.

Since the mid-1950's, much attention has been devoted to problems of finding the shortest route. Recent publications have discussed procedures for solving such problems by hand (Mandt 1973, 1974) and by large, digital computers (Elsner et al. 1975, Martin 1963, Schnelle 1972). The Forest Service has recently adopted a network analysis system (Sullivan 1974), which is perhaps the most flexible system in existence, but it requires the availability of a very large computing facility.

The programs discussed in this paper fall into the middle ground between procedures intended for hand computation and those developed for large-scale computers. These programs were developed and tested on a Hewlett-Packard 9830 programable desk-top calculator and were designed to take advantage of the digitizer-plotter capability of that calculating system. The shortest path program itself, however, is quite general and could be used, without significant revision, on any calculator or computer that uses the standard ASCII (American Standard Code for Information Interchange) BASIC programing language.

Network Analysis

A transportation system (such as a system of forest roads) may be described as a *network*, a collection of interconnected segments or *links*. Each link describes a unique path between two adjacent *nodes*. A node is any feature that might be treated as the point of departure or destination of some path through the network, such as a landing or mill. Nodes may also be road intersections, viewpoints, scaling stations, and bridges. Nodes are also commonly used to indicate points at which road design standards change or there is a marked change in grade or curvature. Such changes would be expected to influence costs of hauling logs (Byrne, et al. 1960) and may therefore be of interest in the solution of many forest transportation problems.

Figure 1 is a map of a small but representative road network for a hypothetical forest. Nodes have been superimposed on the network at each landing, at two grade breaks, at one point at which the road standard changes, at a bridge, at each of the three road intersections, and at the destination (the mill). Numbering of these nodes is arbitrary. Each link, however, is referenced according to the adjacent nodes which it interconnects. Furthermore, links are commonly bidirectional. That is, link 1-2 is "different" from link 2-1, even though each represents the same segment of road. This is a useful distinction for differentiating the cost of hauling logs over an adverse grade in one direction and hauling them over a favorable grade in the opposite direction along the same link.

SHORTEST-PATH PROBLEMS

The specific problem addressed in this paper is that of finding the shortest path through a network from a specified point of departure. The unit of measure selected to judge a path's length can be almost anything. Common units are hauling costs, distance, or time; but other units, such as construction cost, maintenance cost, or even a measure of the scenic or esthetic value along the link, could be used. Although this problem would normally be solved for the shortest path between a specified point of departure and a

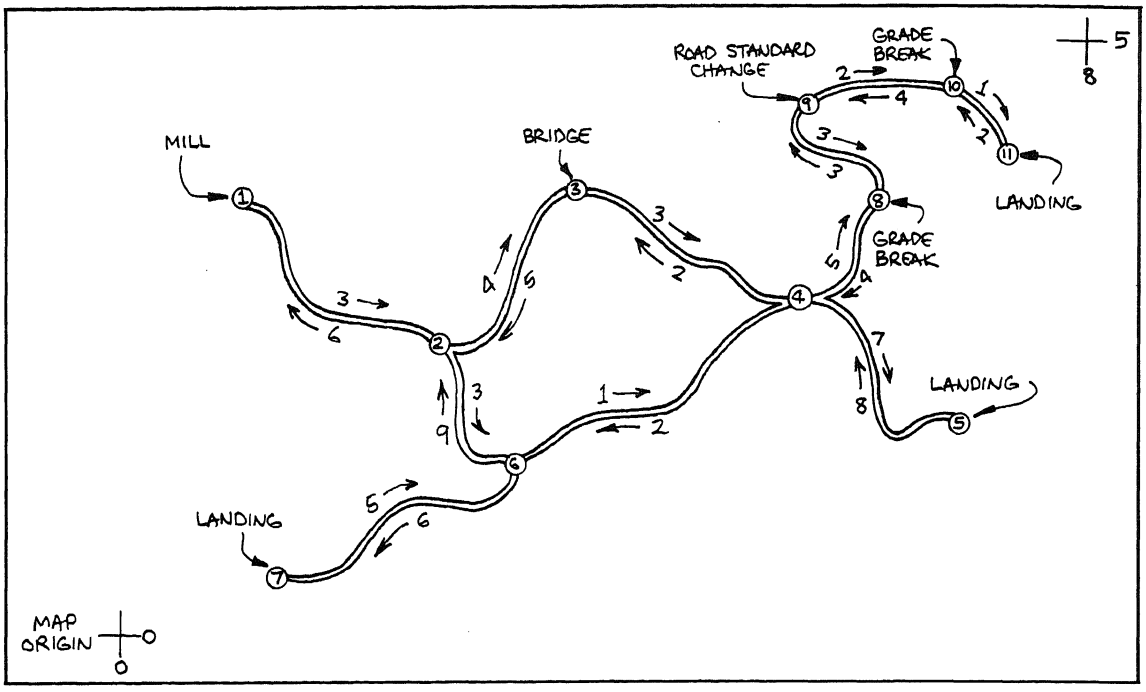


Figure 1.--Source map for the example problem.

specified destination, it may occasionally be of interest to solve the problem for all the shortest paths from the departure node to all other nodes. As an example, we might use this capability to solve for the best network of backhaul routes (i.e., when a truck is traveling empty) from the mill to all landings. The problem of finding all the shortest paths from a departure node to all other nodes is called the "n-shortest-paths problem." It seems that this problem would be much harder to solve than the simple shortest path problem, and several of the faster algorithms for solving the shortest path problem also find the n shortest paths at the same time (Bradley 1975). The solution procedure we have selected belongs to this class of algorithms. It is called the "Moore algorithm" and has been described fully by Martin (1963). Although it always determines the n shortest paths from a specified point of departure, our program permits the user to decide whether all n paths or only the single shortest

Network Analysis Programs

The programs (to be used together) are of two types: three programs are of the first type; they essentially set up the problem by generating the network. The second type (the fourth program, "Optimization Program") solves the problem. The network generator programs are machine dependent; that is, they were developed specifically for the digitizing, plotting, and computing capabilities of the Hewlett-Packard 9830 system. Numerous other systems, however, have similar capabilities; the description in this paper is intended to be general enough that the programs can be converted to other systems.

PROGRAM LIMITATIONS

In their present form the programs can accommodate a maximum of 60 nodes and 255 links, with no more than 8 links

limitations are a function of the calculator memory and could be increased if memory capacity were expanded. The calculator systems on which the programs have been developed and tested have the following configuration:

HP-9830 calculator with 15,808 bytes (7,904 words) of read-write memory

Additional read-only memories:
- plotter control
- extended input/output
- string variable

Flatbed plotter (9862A)
Digitizer (9864A)
Thermal page printer (9866A)

NETWORK GENERATOR

The most burdensome phase in network analysis by computer is usually data entry. Nodes and links must be identified and link values must be assigned before the analysis can begin. A primary advantage of the approach reported here is that the network definition task has been automated by the use of the HP-9830 digitizer.

Three network generator programs are provided for entering problem data into the calculator: one is loaded into the main memory of the calculator, and the other two are loaded into special function keys f_0 and f_1. Table 1 summarizes the input procedure for the data required when these programs are executed. Note that after all three programs have been loaded into the calculator, the program in main memory is executed first. This provides the user with some preliminary instructions and initializes certain internal memory locations. The program on special function key f_0, which enters the network via the digitizer and reproduces it on the plotter, is executed next. This program also computes and applies a "rotation factor" so that the user need not be concerned about whether the source map is correctly alined on the digitizer. After the network has been completely digitized, the program on special function key f_1 is executed. This program establishes a "link matrix" which identifies individual links by their "from" and "to" nodes. It also prints a link summary and permits the user to store the link matrix on a cassette tape if desired.

Figures 2, 3, and 4 summarize the printed output resulting from running the three programs listed above. This output is for the example program in figure 1, entered as shown in table 1. It is important to note that these tables contain printed output only. Numerous requests for data and certain instructions to the user are displayed (in the calculator display window) while these programs are running but are not printed. This convention produces a "clean" printout. A complete track of all printed and displayed messages, as well as a record of all inputs, can be obtained if desired by putting the calculator into "PRT ALL" mode before running the program.

In addition to the printed output, the program on key f_0 also plots the digitized network on the flatbed plotter (fig. 5). This plot is useful as a visual check to insure that the network has been correctly entered into the calculator.

SPECIAL CONSIDERATIONS

It should be evident from the example (fig. 1) that it is often necessary to re-digitize nodes which have already been digitized in order to correctly describe all routes of the network. As an example, consider nodes 4 and 5. Node 5 must be digitized after node 4, and then node 4 must be redigitized in order to continue the network. When this happens, the calculator is programed to sense that the node has already been digitized, and a message to that effect is printed (fig. 3).

Table 1--*Input explanation for network generator programs*

Visual prompt on display or printer (oral prompt by digitizer)	Keyboard or digitizer response	Explanation
	SCRATCH A.	Clear calculator memory.
	LOAD 5.	Load programs from tape.
	LOADKEY 4.	
	RUN.	Begin program execution.
DOCUMENT HORIZ.DIM. (INCHES)?	8.	Enter map dimensions.
DOCUMENT VERT.DIM. (INCHES)?	5.	
SELECT FUNCTION KEY (F0)	f_0.	Depress special function key f_0 to digitize network.
SET THE ORIGIN AT POINT 0,0 ON THE NETWORK SOURCE DOCUMENT (digitizer "beep")	Depress red "0" button on cursor.	Set point to which nodes will be referenced.
DIGITIZE POINT 8,5 ("beep")	Digitize upper right-hand corner of map.	Use "S" button on cursor.
START DIGITIZING NODE LOCATIONS BEGINNING WITH YOUR CHOICE FOR NODE #1		
("beep")	Digitize node 1.	Use "S" button on cursor to enter node locations.
("beep")	Digitize node 2.	
("beep")	Digitize node 3.	
("beep")	Digitize node 4.	
("beep")	Digitize node 5.	
("beep")	Redigitize node 4.	
("beep")	Digitize node 6.	
("beep")	Redigitize node 2.	Define link between node 2 and node 6.
("beep")	Redigitize node 6.	
("beep")	Digitize node 7.	
("beep")	Digitize map origin.	Use "S" button, not '0'; this informs the calculator that a new route is to be entered.
	.	
("beep")	Redigitize node 4.	Enter new route, starting at node 4.
("beep")	Digitize node 8.	
("beep")	Digitize node 9.	
("beep")	Digitize node 10.	
("beep")	Digitize node 11.	
("beep")	STOP.	Halt digitizer program.
	f_1.	Depress special function key f_1 to print a link assignment summary.
. . . (link matrix summary is printed)		
LINK MATRIX COMPLETE. STORE?	YES.	Indicate that the link matrix is to be stored on tape.
FILE # (MINIMUM SIZE =768)?	0.	Store link matrix on tape file 0.
. . . (program terminates)		

```
 THIS SET OF PROGRAMS IS DESIGNED TO DIGITIZE A SET OF NODE
LOCATIONS OFF A SOURCE DOCUMENT. AS EACH IS DIGITIZED IT IS
PLOTTED, NUMBERED, AND STORED IN THE CALCULATOR. THIS IS
ACCOMPLISHED WITH A PROGRAM ON FUNCTION KEY -F0-.
 AFTER THE NODE RECORDING IS COMPLETE,LINK NUMBERING IS DONE BY
A PROGRAM STORED ON FUNCTION KEY -F1-.

 THE FIRST STEP IS TO INFORM THE PLOTTER OF THE SOURCE DOCUMENT
SIZE .

 WAIT FOR THE -SELECT FUNCTION KEY(F0)- MESSAGE,THEN PROCEED.
```

Figure 2.--Printed output from the program in main memory (loaded from tape file 5) for the example problem.

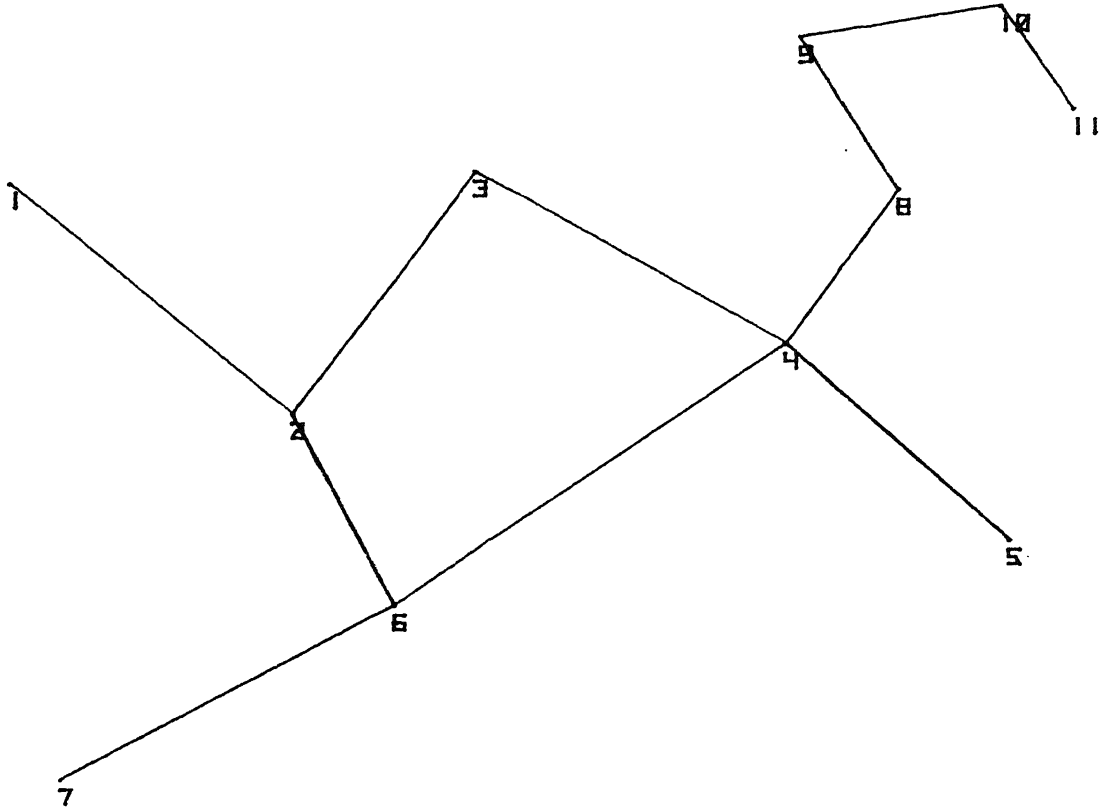
```
SET THE ORIGIN AT POINT 0,0 ON THE NETWORK
     SOURCE DOCUMENT WITH THE DIGITIZER.


DIGITIZE POINT 8     : 5

START DIGITIZING NODE LOCATIONS BEGINNING WITH YOUR CHOICE FOR NOD



NODE AT   1.07,   3.70 ; IDENTIFIED AS #   1
NODE AT   2.78,   2.45 ; IDENTIFIED AS #   2
NODE AT   3.96,   3.73 ; IDENTIFIED AS #   3
NODE AT   5.82,   2.79 ; IDENTIFIED AS #   4
NODE AT   7.15,   1.72 ; IDENTIFIED AS #   5
                                    RECORDED EARLIER AS #
NODE AT   3.43,   1.42 ; IDENTIFIED AS #   6
                                    RECORDED EARLIER AS #
                                    RECORDED EARLIER AS #
NODE AT   1.37,   0.48 ; IDENTIFIED AS #   7
                                     START NEW ROUTE.
                                   BEGIN RESTARTS AT A NO
                                 THAT WAS IDENTIFIED EARL

                                    RECORDED EARLIER AS #
NODE AT   6.47,   3.62 ; IDENTIFIED AS #   8
NODE AT   5.90,   4.45 ; IDENTIFIED AS #   9
NODE AT   7.08,   4.61 ; IDENTIFIED AS #  10
NODE AT   7.54,   4.01 ; IDENTIFIED AS #  11
```

**********LINK ASSIGNMENT SUMMARY**********

LINK#	FROM NODE#	TO NODE#	VALUE1	VALUE2	VALUE3
1	1	2			
2	2	1			
3	5	2			
4	2	6			
5	2	3			
6	3	2			
7	3	4			
8	4	3			
9	5	4			
10	4	5			
11	4	6			
12	6	4			
13	6	7			
14	7	6			
15	4	8			
16	8	4			
17	8	9			
18	9	8			
19	9	10			
20	10	9			
21	10	11			
22	11	10			

(These columns are available as scratch space. They are intended for the user to write in values, such as cost, distance, or time, which may be associated with each link.)

Figure 4.--Printed output from the program on special function key f_1 (loaded from tape file 4) for the example problem.



Occasionally it may be desirable to enter an entirely new route to avoid redigitizing several points. As an example, refer to the dead-end route leading from node 4 to node 11 (fig. 1). Rather than digitizing nodes 4, 8, 9, 10, and 11 and then retracing the route back to node 4, it is convenient to enter those nodes as a new route. This is done after all other nodes have been digitized. For example, the last node to be entered is node 7 (figs. 1 and 3). Then, the fact that a new route is to be entered is signaled by redigitizing the map origin (lower left-hand corner), followed by the nodes of the new route (4, 8, 9, 10, and 11). Any number of new routes can be entered in this manner.

OPTIMIZATION PROGRAM

After the network has been fully described, the calculator memory is cleared and the optimization program is loaded into main memory to solve the problem. The link matrix created with the network generator may then be loaded from a tape (if it was stored as indicated at the bottom of table 1). Otherwise, it can be entered at the keyboard. Although the latter capability may appear unnecessary, it has been included to permit users without digitizers to use the optimization program. Such users would simply enter the link matrix at the keyboard without having previously run the network generator programs.

Once the link matrix has been entered (either from a tape or through the keyboard), values must be associated with each link. These values can also be entered from a tape or through the keyboard. Link values might have been stored on tape after a previous run of the optimization program; normally, they will be entered at the keyboard. As shown in table 2, the calculator prompts the user by

user is given the opportunity to cha any of the link values, to add links or to delete links. Either the link matrix or the value array or both ma then be stored on tape if desired (see table 2).

The optimization routine begins after the user identifies the initia node, or point of departure (table 2 As discussed earlier, this routine finds the shortest paths from the initial node to all other nodes. Th method used for the optimization is not discussed in this paper; interes readers are referred to a complete discussion of the Moore algorithm (Martin 1963).

When the optimization is comple the calculator requests the number of the terminal node (see table 2). The user may then indicate a specifi node, in which case the shortest pat from the initial node to that termin node will be printed; or he may requ that the shortest paths to all nodes be printed.

After these results have been summarized, the user is permitted to run a new optimization from a new initial node. He may also enter a new set of values, either from a tap or through the keyboard.

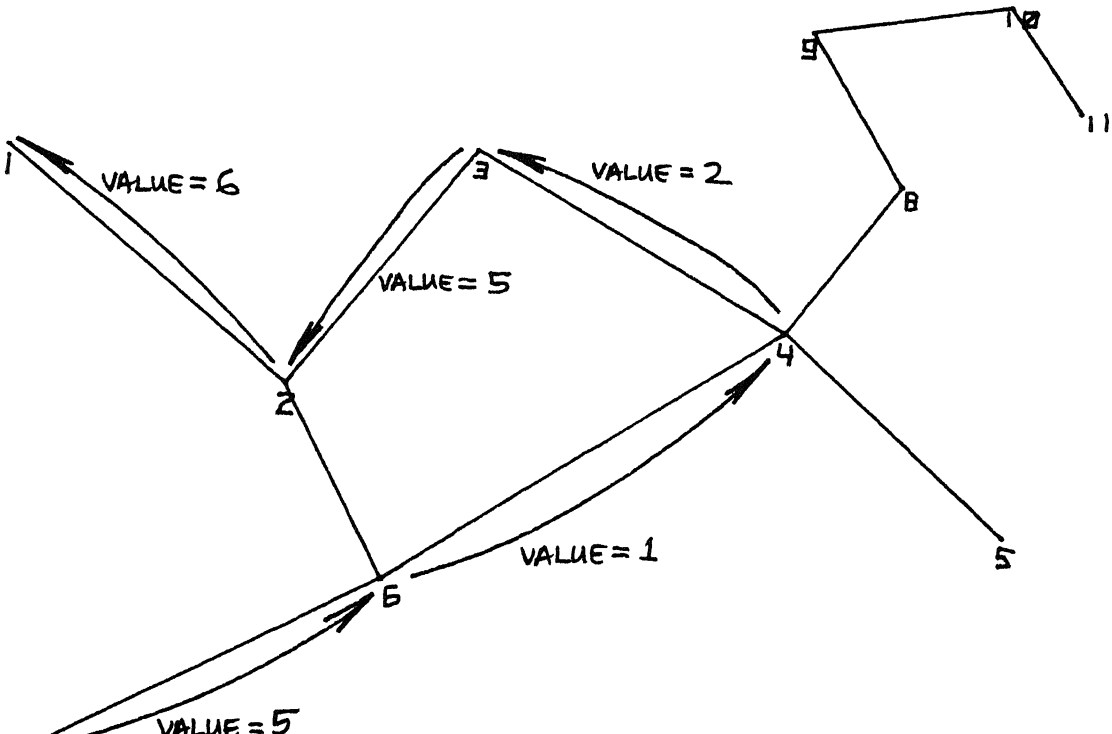
Figure 6 is a listing of the printed output from the optimization program for the example problem. Fo the minimum path analysis, node 7 (a landing) was selected as the poin of departure and node 1 (the mill) a the destination. The "optimum path" from node 7 to node 1 is via nodes 6, 4, 3, and 2 (see fig. 7). The total estimated "cost" of following this path is 19.

Table 2--*Input explanation for the shortest path program*

Visual prompt on display or printer	Keyboard response	Explanation
	SCRATCH A	Clear calculator memory.
	LOAD 6	Load program from tape.
	RUN	Begin program execution.
LOAD LINK MATRIX OFF TAPE?	YES	The link matrix was previously stored on tape; prepare to retrieve it.
FILE # OF LINK MATRIX STORAGE?	0	Identify tape file number.
LOAD VALUES OFF TAPE?	NO	The link values have not previously been stored.
VALUE FOR LINK #1?	3	Enter values corresponding to the links as numbered in figure 4.
VALUE FOR LINK #2?	6	
VALUE FOR LINK #3?	9	
VALUE FOR LINK #4?	3	Although the link values used in this example are whole numbers, decimal numbers may also be entered.
VALUE FOR LINK #5?	4	
VALUE FOR LINK #6?	5	
VALUE FOR LINK #7?	3	Note that the values used here correspond to those in figure 1.
VALUE FOR LINK #8?	2	
VALUE FOR LINK #9?	8	
VALUE FOR LINK #10?	7	
VALUE FOR LINK #11?	2	
VALUE FOR LINK #12?	1	
VALUE FOR LINK #13?	6	
VALUE FOR LINK #14?	5	
VALUE FOR LINK #15?	5	
VALUE FOR LINK #16?	4	
VALUE FOR LINK #17?	3	
VALUE FOR LINK #18?	3	
VALUE FOR LINK #19?	2	
VALUE FOR LINK #20?	4	
VALUE FOR LINK #21?	1	
VALUE FOR LINK #22?	2	
WANT TO CHANGE ANY LINK VALUES?	NO	This option is provided for correcting wrong entries; in this example, no corrections are necessary.
WANT TO ADD A LINK?	NO	
WANT TO ERASE A LINK?	NO	These options are provided for making corrections or revising the network; here, no corrections or revisions are necessary.
STORE LINK MATRIX?	NO	The link matrix has not been changed, and it is already stored on file 0.
STORE VALUES?	YES	Indicate that the link values are to be stored.
FILE # (NOTE: SIZE MUST BE >512)?	1	Store link values on file 1.
*******MINIMUM PATH ANALYSIS		
INITIAL NODE IN PATH?	7	Node 7 (a landing) is selected as the point of departure.
TERMINAL NODE (999=ALL NODES)?	1	Node 1 (the mill) is selected as the destination; if 999 had been entered, the minimum paths from node 7 to all nodes would have been printed.
. . . (the minimum path summary is printed here). . . .		

```
***LINK SUMMARY FOR SYSTEM WITH  11   NODES AND  22   LINKS.
LINK#      FROM NODE#      TO NODE# :          VALUE
  1                1                2              3
  2                2                1              6
  3                6                2              9
  4                2                6              3
  5                2                3              4
  6                3                2              5
  7                3                4              3
  8                4                3              8
  9                5                4              8
  10               4                5              7
  11               4                6              2
  12               6                4              1
  13               6                7              6
  14               7                6              5
  15               4                8              5
  16               8                4              4
  17               8                9              3
  18               9                8              3
  19               9                10             2
  20               10               9              4
  21               10               11             1
  22               11               10             2
******* MINIMUM PATH ANALYSIS.

MINIMUM PATH FROM NODE  7    TO NODE  1    HAS VALUE  19
THE NODE-TO-NODE PATH IS  7
                          6
                          4
                          3
                          2
                          1
```

Figure 6.--Printed output from the shortest path program (loaded from tape file 6) for the example problem.



Concluding Remarks

The network analysis package described in this paper is intended for use with the small- to intermediate-size networks commonly encountered in planning transportation in forests. It is not suited to the analysis of large networks. Probably the most attractive feature of this package is the network generation capability which permits rapid, dependable entry of the network via a digitizer/plotter system. Also attractive is the interactive, problem-solving feature of the programs, made possible by the stand-alone capability of the desk-top programable calculator system.

Literature Cited

Bradley, G. H.
1975. Survey of deterministic networks. Am. Inst. Ind. Eng. Trans. 7(3):222-234.

Byrne, J. J., R. J. Nelson, and P. H. Googins.
1960. Logging road handbook: the effect of road design on hauling costs. Agric. Handb. No. 183, 65 p. U.S. Dep. Agric. For. Serv., Washington, D.C.

Elsner, G. H., M. R. Travis, and P. H. Kourtz.
1975. Dynamic programming subroutines based on the Dijkstra algorithm for finding minimum cost paths in directed networks. Inf. Rep. FF-X-51, 16 p. For. Fire Res. Inst., Can. For. Serv., Ottawa, Ont.

Mandt, C. I.
1973. Network analysis in transportation planning. *In* Planning and decisionmaking as applied to forest harvesting: symposium proceedings, Corvallis, Oreg.,

Mandt, C. I.
1974. A longhand approach to resource transportation analysis using a link-node network. Eng. Tech. Inf. Syst. Tech. Rep. ETR-7700-4e, 28 p. USDA For. Serv., Washington, D.C.

Martin, B. V.
1963. Minimum path algorithms for transportation planning. Res. Rep. R63-52, 105 p. Highw. Transp. Demand Res. Proj., Sch. Eng., Mass. Inst. Technol.

Schnelle, W. F.
1972. Application of network analysis to National Forest transportation problems. Field Notes 4(3,4):1-12. USDA For. Serv., Eng. Tech. Inf. Syst., Washington, D.C.

Sullivan, E. C.
1974. Network analysis user's guide. Spec. Rep. Inst. Transp. and Traffic Eng., Univ. Calif., Berkeley.

Appendix

Listing of the Network
Generator Initialization Program

```
 DIM M[2],NI[60,10],LI[256,3]
 REM NETWORK BUILDER.
 DIM A$[30]
 PRINT TAB5"THIS SET OF PROGRAMS IS DESIGNED TO DIGITIZE A SET OF NODE"
 PRINT TAB4"LOCATIONS OFF A SOURCE DOCUMENT. AS EACH IS DIGITIZED IT IS "
 PRINT TAB4"PLOTTED, NUMBERED, AND STORED IN THE CALCULATOR. THIS IS"
 PRINT TAB4"ACCOMPLISHED WITH A PROGRAM ON FUNCTION KEY -F0-."
 PRINT TAB5"AFTER THE NODE RECORDING IS COMPLETE,LINK NUMBERING IS DONE BY
 PRINT TAB4"A PROGRAM STORED ON FUNCTION KEY -F1-. "TAB100
0 PRINT TAB5"THE FIRST STEP IS TO INFORM THE PLOTTER OF THE SOURCE DOCUMEN
0 PRINT TAB4"SIZE ."TAB100
0 M[1]=M[2]=0
0 DISP "DOCUMENT HORZ.DIM.(INCHES)";
0 INPUT G1
0 DISP "DOCUMENT VERT.DIM.(INCHES)";
0 INPUT G2
0 SCALE 0,G1*100,0,G2*100
0 PRINT TAB5"WAIT FOR THE -SELECT FUNCTION KEY(F0)- MESSAGE,THEN PROCEED."
0 PRINT TAB100
0 REM ********** INITIALIZE THE NODE MATRIX.
0 FOR I=1 TO 60
0 FOR J=3 TO 10
0 N[I,J]=0
0 NEXT J
0 NEXT I
0 DISP "SELECT FUNCTION KEY(F0)";
0 STOP
0 END
```

Listing of the Network Generator Program
on Special Function Key f_0

```
10 REM NETWORK BUILDER#1: FOR ENTERING,NUMBERING AND STORING CONNECTED
20 PLOT 0,0,1
30 DIM A$[3]
40 PRINT TAB100"SET THE ORIGIN AT POINT 0,0 ON THE NETWORK"
50 PRINT TAB5"SOURCE DOCUMENT WITH THE DIGITIZER."TAB100
60 WRITE (9,*)
70 WAIT 5000
80 PRINT TAB100"DIGITIZE POINT"G1","G2,TAB100
90 WRITE (9,*)
100 ENTER (9,*)X1,Y1
110 REM ********** COMPUTE ROTATION FACTORS.
120 S8=SIN(ATN((Y1-G2)/X1))
130 C8=COS(ATN((Y1-G2)/X1))
140 PRINT "START DIGITIZING NODE LOCATIONS BEGINNING WITH YOUR CHOICE F
150 PRINT TAB100,TAB100
160 M[1]=I1=1
170 WRITE (9,*)
180 ENTER (9,*)X1,Y1
190 REM  ROTATE COORDINATES.
200 X2=X1*C8+Y1*S8
210 Y1=Y1*C8-X1*S8
220 X1=X2
230 N[1,1]=X1*100
240 N[1,2]=Y1*100
250 WRITE (15,260)X1,Y1,I1
260 FORMAT "NODE AT",F6.2,",",F6.2," ; IDENTIFIED AS #",F3.0
270 PLOT N[I1,1],N[I1,2],0
280 CPLOT -1,-1
290 LABEL (*)I1
300 GOTO 440
310 REM SUBSEQUENT ROUTE RECORDING STARTS AT THE NEXT ENTER.
320 PRINT TAB44"BEGIN RESTARTS AT A NODE"
330 PRINT TAB42"THAT WAS IDENTIFIED EARLIER."TAB100
340 WRITE (9,*)
350 ENTER (9,*)X1,Y1
360 REM   ROTATE COORDINATES.
370 X2=X1*C8+Y1*S8
380 Y1=Y1*C8-X1*S8
390 X1=X2
400 GOSUB 840
410 IF J1=M[1]+1 THEN 320
420 PRINT TAB45"RECORDED EARLIER AS #"J1
430 I1=J1
440 REM CONTINUE DIGITIZING REMAINING POINTS IN ROUTE.
450 FOR K=M[1] TO 1000
460 WRITE (9,*)
470 ENTER (9,*)X1,Y1
480 X2=X1*C8+Y1*S8
490 Y1=Y1*C8-X1*S8
500 X1=X2
```

```
510 REM CHECK TO SEE WHETHER ORIGIN HAS BEEN DIGITIZED.
520 IF SQR(X1*X1+Y1*Y1)>0.1 THEN 550
530 PRINT TAB45"START NEW ROUTE."
540 GOTO 360
550 J1=M[1]+1
560 GOSUB 840
570 IF J1=M[1]+1 THEN 610
580 PRINT TAB45"RECORDED EARLIER AS #"J1
590 M[1]=M[1]-1
600 GOTO 670
610 WRITE (15,260)X1,Y1,J1
620 N[J1,1]=X1*100
630 N[J1,2]=Y1*100
640 PLOT N[J1,1],N[J1,2],0
650 CPLOT -1,-1
660 LABEL (*)J1
670 REM ESTABLISH WHETHER THE I1 NODE FOR THIS LINK HAS BEEN RECORDED EARL
680 FOR J=3 TO 10
690 IF N[J1,J]=I1 THEN 790
700 IF N[J1,J]#0 THEN 740
710 REM THE I1 NODE HAS NOT BEEN RECORDED EARLIER.
720 N[J1,J]=I1
730 GOTO 770
740 NEXT J
750 PRINT "TOO MANY LINKS EMINATING FROM NODE#"I1". LIMIT IS 8."
760 END
770 PLOT N[I1,1],N[I1,2],-2
780 PLOT N[J1,1],N[J1,2],-1
790 I1=J1
800 M[1]=M[1]+1
810 NEXT K
820 PRINT "NUMBER OF NODES EXCEEDS LIMIT OF 60 SET IN THE COM STATEMENT."
830 STOP
840 REM***SUB1***ROUTINE TO CHECK WHETHER POINT HAS BEEN IDENTIFIED EARLIE
850 FOR J1=1 TO M[1]
860 IF SQR((N[J1,1]-X1*100)↑2+(N[J1,2]-Y1*100)↑2)<10 THEN 890
870 REM IF NODE WAS NOT IDENTIFIED EARLIER THIS ROUTINE RETURNS J1=M(1)+1.
880 NEXT J1
890 RETURN
900 END
```

Listing of the Link Matrix Generator Program
on Special Function Key f_1

```
REM NETWORK BUILDER #2:PROGRAM DESIGNED TO SET UP LINK MATRIX.
DIM A$[3]
REM BUILD THE LINK MATRIX AND PRINT OUT A SUMMARY.
L1=0
PRINT TAB100"**************LINK ASSIGNMENT SUMMARY************"
PRINT "LINK#      FROM NODE#      TO NODE# :     VALUE1        VALUE2      VALU
REM GO THROUGH ALL ENTRIES IN THE N MATRIX.
FOR J1=1 TO M[1]
FOR J=3 TO 10
 IF N[J1,J]=0 THEN 330
 IF N[J1,J]=-1 THEN 320
 I1=N[J1,J]
 L1=L1+1
 L[L1,3]=I1*100+J1
 L[L1,1]=N[J1,1]
 L[L1,2]=N[J1,2]
 PRINT L1,I1,J1
 REM HAS THE REVERSE LINK BEEN RECORDED?
 FOR I=3 TO 10
 IF N[I1,I]=J1 THEN 250
 IF N[I1,I]#0 THEN 240
 REM THE REVERSE LINK MUST NOT HAVE BEEN BEEN RECORDED.  WE DO SO.
 GOTO 270
 NEXT I
 REM THE REVERSE LINK WAS RECORDED PREVIOUSLY.
 N[I1,I]=-1
 L1=L1+1
 L[L1,3]=J1*100+I1
 L[L1,1]=N[I1,1]
 L[L1,2]=N[I1,2]
 PRINT L1,J1,I1
 NEXT J
 NEXT J1
 DISP "LINK MATRIX COMPLETE. STORE";
 INPUT A$
 IF A$="YES" THEN 380
 STOP
 REM PREPARE TO STORE.
 REWIND
 DISP "FILE#(MINIMUM FILE SIZE=768)";
 INPUT F9
 L[256,1]=L1
 L[256,2]=M[1]
 STORE  DATA F9,L
 REWIND
 END
```

Listing of the Shortest-path Program

```
10 REM NETWORK PROGRAM: N-SHORTEST PATHS ROUTINE (MOORE ALGORITHM).
20 DIM L[256,3],V[256],J[60],S[60],N[60],P[60],A$[3],D[10,2]
30 REWIND
40 F8=F9=I9=L9=0
50 DISP "LOAD LINK MATRIX OFF TAPE";
60 INPUT A$
70 IF A$="NO" THEN 340
80 DISP "FILE# OF LINK MATRIX STORAGE";
90 INPUT F9
100 LOAD  DATA F9,L
110 L1=L[256,1]
120 N1=L[256,2]
130 DISP "LOAD VALUES OFF TAPE";
140 INPUT A$
150 IF A$="NO" THEN 220
160 DISP "FILE# OF VALUE STORAGE";
170 INPUT F8
180 IF F8>F9 THEN 200
190 REWIND
200 LOAD  DATA F8,V
210 I9=1
220 PRINT "***LINK SUMMARY FOR SYSTEM WITH "N1"NODES AND "L1"LINKS."
230 PRINT "LINK#      FROM NODE#      TO NODE# :          VALUE"
240 FOR K=1 TO L1
250 I1=INT(L[K,3]/100)
260 T1=L[K,3]-I1*100
270 IF I9=1 THEN 300
280 DISP "VALUE FOR LINK#"K;
290 INPUT V[K]
300 PRINT K,I1,T1,V[K]
310 NEXT K
320 IF L9#0 THEN 590
330 GOTO 550
340 PRINT "KEYBOARD ENTRY MODE."
350 DISP "NUMBER OF NODES";
360 INPUT N1
370 DISP "NUMBER OF LINKS";
380 INPUT L1
390 PRINT "***LINK SUMMARY FOR SYSTEM WITH "N1"NODES AND "L1"LINKS."
400 PRINT "LINK#      FROM NODE#      TO NODE# :          VALUE"
410 L[256,1]=L1
420 L[256,2]=N1
430 FOR K=1 TO L1
440 DISP "INITIAL NODE FOR LINK#"K;
450 INPUT I1
```

```
480 L[K,3]=I1*100+T1
490 DISP "VALUE FOR LINK"K;
500 INPUT V[K]
510 I1=INT(L[K,3]/100)
520 T1=L[K,3]-I1*100
530 PRINT K,I1,T1,V[K]
540 NEXT K
550 REM ZERO THE REMAINDER OF THE MATRICES L AND V.
560 FOR K=L1+1 TO 256
570 L[K,3]=V[K]=0
580 NEXT K
590 DISP "WANT TO CHANGE ANY LINK VALUES";
600 INPUT A$
610 IF A$="NO" THEN 690
620 DISP "LINK # AND NEW VALUE";
630 INPUT L9,V[L9]
640 DISP "MORE CHANGES";
650 INPUT A$
660 IF A$="YES" THEN 620
670 PRINT TAB100,TAB100
680 GOTO 210
690 DISP "WANT TO ADD A LINK";
700 INPUT A$
710 IF A$="NO" THEN 890
720 PRINT "LINK#     FROM NODE#     TO NODE# :          VALUE"
730 L1=L1+1
740 DISP "INITIAL NODE FOR LINK#"L1;
750 INPUT I1
760 DISP "TERMINAL NODE FOR LINK#"L1;
770 INPUT T1
780 L[L1,3]=I1*100+T1
790 DISP "VALUE FOR LINK"L1;
800 INPUT V[L1]
810 I1=INT(L[L1,3]/100)
820 T1=L[L1,3]-I1*100
830 PRINT L1,I1,T1,V[L1]
840 L[256,1]=L1
850 IF I1 <= N1 AND T1 <= N1 THEN 690
860 N1=N1+1
870 L[256,2]=N1
880 GOTO 690
890 DISP "WANT TO ERASE A LINK";
900 INPUT A$
910 IF A$="NO" THEN 970
920 DISP "LINK #";
930 INPUT L9
940 V[L9]=1E+05
950 PRINT "LINK#"L9"HAS BEEN 'ERASED' BY ASSIGNING IT THE VALUE 1
960 GOTO 890
970 DISP "STORE LINK MATRIX";
980 INPUT A$
990 IF A$="NO" THEN 1030
1000 DISP "FILE#(NOTE:FILE MUST BE>768)";
1010 INPUT F9
1020 STORE  DATA F9,L
```

```
1030 DISP "STORE VALUES";
1040 INPUT A$
1050 IF A$="NO" THEN 1110
1060 DISP "FILE#(NOTE:FILE MUST BE>512)";
1070 INPUT F8
1080 IF F8>F9 THEN 1100
1090 REWIND
1100 STORE  DATA F8,V
1110 REWIND
1120 PRINT "******* MINIMUM PATH ANALYSIS."
1130 REM INITIALIZE SEQUENCE TABLE.****1.
1140 FOR K=1 TO 20
1150 N[K]=0
1160 T[K]=1E+63
1170 NEXT K
1180 REM INITIALIZE SUM TABLE.
1190 FOR K=1 TO 60
1200 J[K]=0
1210 S[K]=1E+63
1220 NEXT K
1230 REM IDENTIFY NODE OF ORIGIN.****2.
1240 DISP "INITIAL NODE IN PATH";
1250 INPUT N0
1260 N5=N0
1270 J[N0]=S[N0]=0
1280 REM SEARCH LINK TABLE FOR 'FROM'NODE.
1290 K1=0
1300 K1=K1+1
1310 I1=INT(L[K1,3]/100)
1320 IF I1=N0 THEN 1350
1330 IF K1=L1 THEN 1440
1340 GOTO 1300
1350 REM ACCUMULATE TIME FROM ORIGIN TO 'TO' NODE.*****3.
1360 T1=L[K1,3]-I1*100
1370 S1=S[N0]+V[K1]
1380 REM COMPARE TEMP SUM S1 TO TREE TABLE SUM FOR THE 'TO'NODE.****4.
1390 IF S1>S[T1] THEN 1410
1400 GOSUB 1610
1410 IF K1=L1 THEN 1440
1420 REM CONTINUE LOOKING FOR 'FROM' NODES.
1430 GOTO 1300
1440 REM CHECK SEQUENCE TABLE FOR ENTRIES.
1450 FOR K=1 TO 20
1460 IF N[K]>0 THEN 1490
1470 NEXT K
1480 GOTO 1520
1490 REM PREPARE TO IDENTIFY NEXT 'FROM' NODE.
1500 GOSUB 1790
1510 GOTO 1280
1520 REM GOTO MIN PATH PRINT OUT.
1530 GOSUB 1940
1540 DISP "OTHER PATHS FROM ANOTHER NODE";
1550 INPUT A$
```

```
1580 INPUT A$
1590 IF A$="YES" THEN 130
1600 GOTO 2210
1610 REM***SUB1**MANIPULATION OF TREE TABLE.
1620 IF S[T1]=1E+63 THEN 1700
1630 M=1
1640 FOR K=1 TO 20
1650 IF S[T1]=T[K] THEN 1670
1660 NEXT K
1670 S[T1]=S1
1680 J[T1]=I1
1690 GOTO 1760
1700 M=0
1710 S[T1]=S1
1720 J[T1]=I1
1730 FOR K=1 TO 20
1740 IF N[K]=0 THEN 1760
1750 NEXT K
1760 T[K]=S1
1770 N[K]=T1
1780 RETURN
1790 REM***SUB2***IDENTIFICATION OF NEXT 'FROM' NODE.
1800 REM FIND MINIMUM VALUE IN SEQUENCE TABLE.
1810 T0=T[1]
1820 N0=N[1]
1830 I2=1
1840 FOR K=1 TO 20
1850 IF T0<T[K] THEN 1890
1860 T0=T[K]
1870 N0=N[K]
1880 I2=K
1890 REM PRINT TAB25;K;N[K];T[K]
1900 NEXT K
1910 T[I2]=1E+63
1920 N[I2]=0
1930 RETURN
1940 REM***SUB3***DETERMINATION AND PRINTING OF MINIMUM PATHS.
1950 DISP "TERMINAL NODE (999=ALL NODES)";
1960 INPUT K4
1970 FOR K=N1 TO 1 STEP -1
1980 IF K#K4 AND K4#999 THEN 2150
1990 K2=K
2000 K1=0
2010 K1=K1+1
2020 N[K1]=J[K2]
2030 IF J[K2]=0 THEN 2060
2040 K2=J[K2]
2050 GOTO 2010
```

```
2060 PRINT TAB100"MINIMUM PATH FROM NODE "N5"TO NODE "K"HAS VALUE "S[
2070 IF K=N5 THEN 2150
2080 FOR K3=(K1-1) TO 1 STEP -1
2090 IF K3#(K1-1) THEN 2120
2100 PRINT "THE NODE-TO-NODE PATH IS "N[K3]
2110 GOTO 2130
2120 PRINT TAB25;N[K3]
2130 NEXT K3
2140 PRINT TAB25;K
2150 NEXT K
2160 IF K4=999 THEN 2200
2170 DISP "ANOTHER TERMINAL NODE";
2180 INPUT A$
2190 IF A$="YES" THEN 1950
2200 RETURN
2210 END
```

☆U.S. GOVERNMENT PRINTING OFFICE: 1978—797-527

The mission of the PACIFIC NORTHWEST FOREST AND RANGE EXPERIMENT STATION is to provide the knowledge, technology, and alternatives for present and future protection, management, and use of forest, range, and related environments.

Within this overall mission, the Station conducts and stimulates research to facilitate and to accelerate progress toward the following goals:

1. Providing safe and efficient technology for inventory, protection, and use of resources.
2. Developing and evaluating alternative methods and levels of resource management.
3. Achieving optimum sustained resource productivity consistent with maintaining a high quality forest environment.

The area of research encompasses Oregon, Washington, Alaska, and, in some cases, California, Hawaii, the Western States, and the Nation. Results of the research are made available promptly. Project headquarters are at:

Fairbanks, Alaska	Portland, Oregon
Juneau, Alaska	Olympia, Washington
Bend, Oregon	Seattle, Washington
Corvallis, Oregon	Wenatchee, Washington
La Grande, Oregon	

Mailing address: *Pacific Northwest Forest and Range Experiment Station*
P.O. Box 3141
Portland, Oregon 97208

The FOREST SERVICE of the U.S. Department of Agriculture is dedicated to the principle of multiple use management of the Nation's forest resources for sustained yields of wood, water, forage, wildlife, and recreation. Through forestry research, cooperation with the States and private forest owners, and management of the National Forests and National Grasslands, it strives — as directed by Congress — to provide increasingly greater service to a growing Nation.

The U.S. Department of Agriculture is an Equal Opportunity Employer. Applicants for all Department programs will be given equal consideration without regard to race, color, sex or national origin.

CARNEGIE LIBRARY
SEP 12 1978
SUPT. DOCS.